What is claimed is:

[c1] 1.A method of employing semaphores to limit access to a shared resource used by a multi-tasking processor, comprising:
providing a first bitmap in a register that prevents specified tasks from running because the specified tasks are awaiting access to an occupied semaphore;
storing an indication in memory that indicates whether the semaphore is occupied;
storing a second bitmap in memory that identifies tasks that are awaiting access to the semaphore; and
attempting to access the semaphore based on checking the indication in memory.

[c2] 2.The method of claim 1, wherein a task checking the indication in memory determines that the semaphore is available, further comprising the steps of setting the indication to indicate that the semaphore is occupied and performing the processing for the task.

[c3] 3.The method of claim 2, wherein performing the processing for the task includes critical section execution.

[c4] 4.The method of claim 3, wherein the critical section includes at least one of external memory accesses and task switches.

[c5] 5.The method of claim 2, further comprising the step of resetting the indication to indicate that the semaphore is available after the step of performing the processing for the task.

[c6] 6.The method of claim 5, further comprising the step of removing from the first bitmap those tasks now included in the second bitmap in memory that identifies tasks that are awaiting access to the semaphore, thereby allowing those tasks to be scheduled for access to the semaphore.

[c7] 7.The method of claim 1, wherein a task checking the indication in memory determines that the semaphore is occupied, further comprising the steps of including the task in the second bitmap and revising the first bitmap to reflect the tasks from the list in the second bitmap.

[c8] 8.The method of claim 7, further comprising the steps of removing the task from the second bitmap when the indication reflects that the semaphore is available and revising the first bitmap to reflect the tasks from the list in the second bitmap, thereby allowing the task to access the semaphore and perform the task processing.

[c9] 9.A system employing semaphores to limit access to a shared resource used by a multi-tasking processor, comprising:
a first bitmap in a register that prevents specified tasks from running because the specified tasks are awaiting access to an occupied semaphore;
an indication in memory that indicates whether the semaphore is occupied;
a second bitmap in memory that identifies tasks that are awaiting access to the semaphore; and
means for attempting to access the semaphore based on checking the indication in memory.

[c10] 10.The system of claim 9, wherein the means for attempting is a processor executing a task.

[c11] 11.The system of claim 10, wherein the task is enabled to access the semaphore when the indication reflects that the semaphore is available.

[c12] 12.The system of claim 10, wherein the task registers itself with the second bitmap and updates the first bitmap when the indication reflects that the semaphore is occupied.

[c13] 13.The system of claim 10, wherein the task execution includes processing a critical section including at least one of external memory accesses and task switching.

[c14] 14.The system of claim 13, wherein the indication in memory is reset to indicate that the semaphore is available after processing the critical section.